

I/O Systems



Exercises

- 12.8 When multiple interrupts from different devices appear at about the same time, a priority scheme could be used to determine the order in which the interrupts would be serviced. Discuss what issues need to be considered in assigning priorities to different interrupts.
- 12.9 What are the advantages and disadvantages of supporting memory-mapped I/O to device-control registers?
- 12.10 Consider the following I/O scenarios on a single-user PC:
- A mouse used with a graphical user interface
 - A tape drive on a multitasking operating system (with no device preallocation available)
 - A disk drive containing user files
 - A graphics card with direct bus connection, accessible through memory-mapped I/O

For each of these scenarios, would you design the operating system to use buffering, spooling, caching, or a combination? Would you use polled I/O or interrupt-driven I/O? Give reasons for your choices.

- 12.11 In most multiprogrammed systems, user programs access memory through virtual addresses, while the operating system uses raw physical addresses to access memory. What are the implications of this design for the initiation of I/O operations by the user program and their execution by the operating system?
- 12.12 What are the various kinds of performance overhead associated with servicing an interrupt?
- 12.13 Describe three circumstances under which blocking I/O should be used. Describe three circumstances under which nonblocking I/O should be used. Why not just implement nonblocking I/O and have processes busy-wait until their devices are ready?

- 12.14** Typically, at the completion of a device I/O, a single interrupt is raised and appropriately handled by the host processor. In certain settings, however, the code that is to be executed at the completion of the I/O can be broken into two separate pieces. The first piece executes immediately after the I/O completes and schedules a second interrupt for the remaining piece of code to be executed at a later time. What is the purpose of using this strategy in the design of interrupt handlers?
- 12.15** Some DMA controllers support direct virtual memory access, where the targets of I/O operations are specified as virtual addresses and a translation from virtual to physical address is performed during the DMA. How does this design complicate the design of the DMA controller? What are the advantages of providing such functionality?
- 12.16** UNIX coordinates the activities of the kernel I/O components by manipulating shared in-kernel data structures, whereas Windows uses object-oriented message passing between kernel I/O components. Discuss three pros and three cons of each approach.
- 12.17** Write (in pseudocode) an implementation of virtual clocks, including the queuing and management of timer requests for the kernel and applications. Assume that the hardware provides three timer channels.
- 12.18** Discuss the advantages and disadvantages of guaranteeing reliable transfer of data between modules in the STREAMS abstraction.